Joint Training Federation Prototype (JTFp) Common-Software Development and Use

Presented to:

9 October 1996

15th AMG Meeting

William F. Waite
AEgis Research Corporation
6703 Odyssey Drive, Huntsville, AL
(205) 922-0802/0904 FAX
BWaite@AEgisRC.com

LT Billy Hudgins
JSIMS/JPO
12249 Science Drive, Ste. 260
(407) 384-5541 / 5599 FAX
hudginsb@jsims.mil

OUTLINE

➡ • JTFp Common Software Functional (and Structural) Requirements

• JTFp Common Software Design

• JTFp Common Software Lessons-Learned

JTFp Common Software Functional Requirements

- ORB OPERATIONS
  - Implement compliant interface
    (Satisfy API, semantics of functional I/F Spec.)
- REPRESENTATIONAL CONSISTENCY
  - Coordinate conversion
- COMPUTATION SERVICES
  - Intervisibility and other environmental effects
- EXERCISE MANAGEMENT
  - Marshal federates to I.C.
  - Instrumentation
- PRESERVE EXECUTION EFFICIENCY

JTFp Common Software Functional Requirements, Cont.

- NETWORK MONITORING
  - Network Snooper
  - Communication Performance Monitor
- INTEGRATED POST PROCESSING
  - On-line After Action Review (AAR)
  - On-line Performance Evaluation
- BROKERAGE
  - Aggregation / dis-aggregation
  - Interest-list management
  - Representation domain interactions (L-V, G-S, C-C, L-R, etc.)

OUTLINE

- JTFp Common Software Requirements
- ➡ JTFp Common Software Design
- JTFp Common Software Lessons-Learned

JTFp Common Software Design Approach Alternatives

- MIDDLEWARE
  - Object Interface
  - Representation consistency
  - Architectural Services (state saving, etc.)
- SERVICE LIBRARIES
- ADJUNCT DATABASES
- ADJUNCT EXECUTIVE FEDERATES
  - Fed.-Controller, -Monitor, Scenario Monitor
- ALLOCATION TO REP. FEDERATE
- BUILT-IN ARTIFACTS
  - Federation Status Object, Data Logger

JTFp Common Software Design Approach

- HYBRID
  - Interface development by federate
  - Adjunct federation-executive components
  - Built-in artifacts
  - Allocation of function to representational federate
- INTERFACE DEVELOPMENT BY FEDERATE
  - Multiple interfaces developed for re-use
  - Variety in packaging interface for comparison
- ADJUNCT FEDERATION-EXECUTIVE COMPONENTS
  - Marshals federation initialization and execution
  - Re-usable and adaptable to other federation

JTFp Common Software Design Approach, Cont.

- BUILT-IN ARTIFACTS
  - Incorporated into FOM
  - Flexible to customize to federation requirements
- ALLOCATION OF FUNCTION TO REPRESENTATIONAL FEDERATE
  - Provides common representation to all federates
  - Single implementation of function

OUTLINE

- JTFp Common Software Requirements

- JTFp Common Software Design

- ➡ JTFp Common Software Lessons-Learned

JTFp Common SW Lessons-Learned - Pros and Cons -

- JTFp DESIGN PROs
  - Met mission requirements
  - Consistent with Collaborative JTFp Team Process and federate constraints
  - Components provided unanticipated utility (e.g. Federation controller served as test driver for system integration) and prospective re-usability
- JTFp DESIGN CONs
  - Required federates to support an additional FOM object and multiple interactions
  - Required federates to perform their own data logging for off-line post processing.

JTFp Common SW Lessons-Learned - Improvement Opportunities -

- AVAILABILITY OF REUSABLE ELEMENTS
  - Artifacts
  - Federation Components
  - Software Designs
  - Service Libraries
- GUIDANCE FOR SOFTWARE ARCHITECTURE TRADES
- USER BINDINGS TO THE RTI

JTFp Common SW Lessons-Learned - Reuse Opportunities -

- ARTIFACTS
  - Federation Status Object
  - Data Loggers' DIF
- COMPONENTS
  - Federation Controller
  - Federation Monitor
  - Scenario Monitor
  - Data Post-Processor
- DESIGNS
  - Federate Interface Shells
  - Language Bindings

JTFp Lessons-Learned - General Observations -

- VARIETY
  - Design electives promote variety
  - Systems engineering determines outcomes
  - Variety implies constraint on A&I substitution of federates among federations
- FEDERATE CONSTRAINTS
  - Development custody and practices
  - Existing internal designs and interfaces
  - Languages (LISP, C++, Smalltalk)
- GOOD NEWS
  - More than one way to successful federations
  - Even independent interface design strategy works
  - Prospects for SW reuse in HLA are positive